*HARSH KASHYAP  
CSE 4*

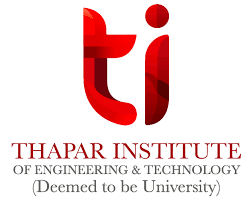
*101917088*

[*hkashyap\_be19@thapar.edu*](mailto:hkashyap_be19@thapar.edu)

A Practical activity Report submitted

for Practical Computing(UCS311)

**PRACTICAL COMPUTING**

****

Computer Science and Engineering

Patiala Campus

**2020**

Submitted to

Aashima Sharma

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Assignment 3**

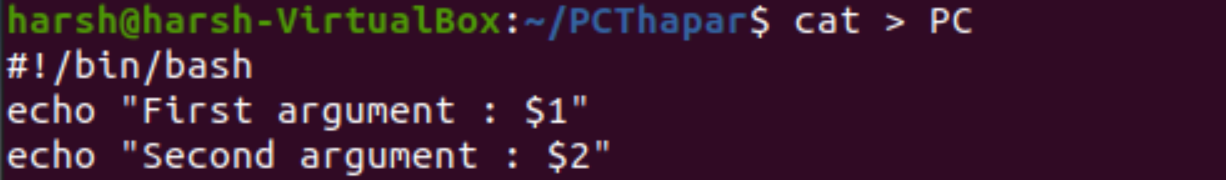
**Question 1**

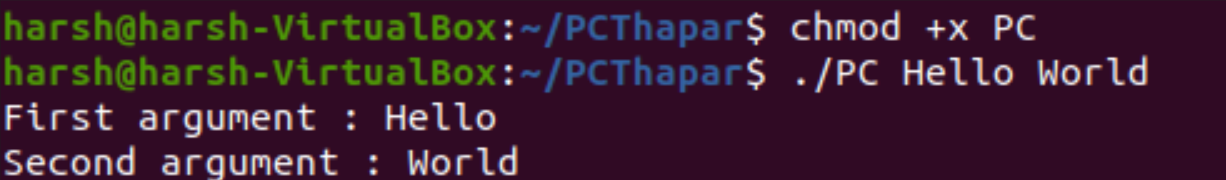
**How can we pass arguments to a script in Linux? And how to access these arguments**

**from within the script?**

**Solution -**

Arguments can be passed to the script when it is executed, by writing them as a space-delimited list following the script file name. Inside the script, the $1 variable references the first argument in the command line, $2 the second argument and so forth. The variable $0 references to the current script.



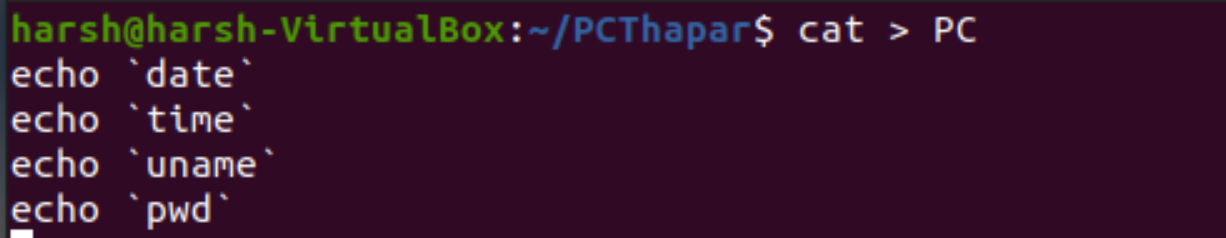
To access the elements, write the argument while you are going to run the script.

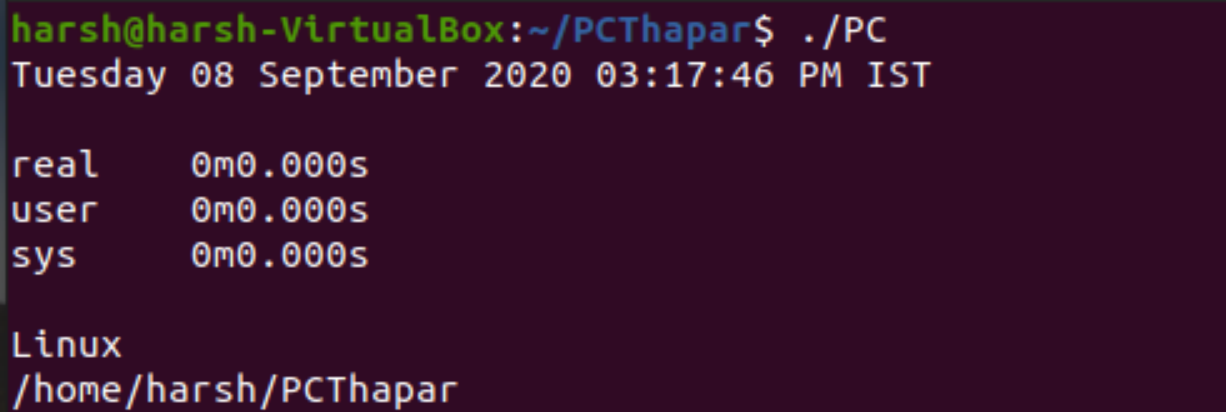
**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Question 2**

**Write a shell script to get the current date, time, username and current working directory.**

**Solution**



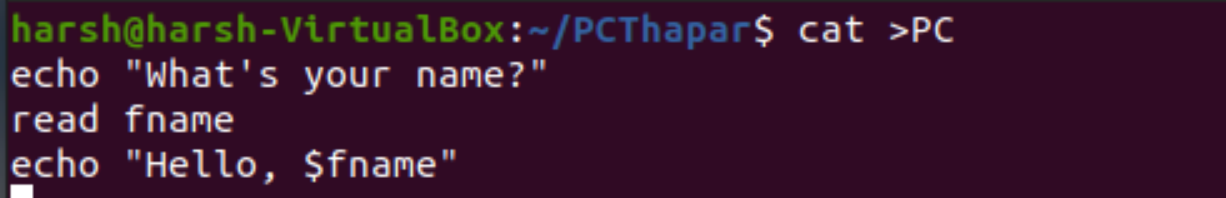
****

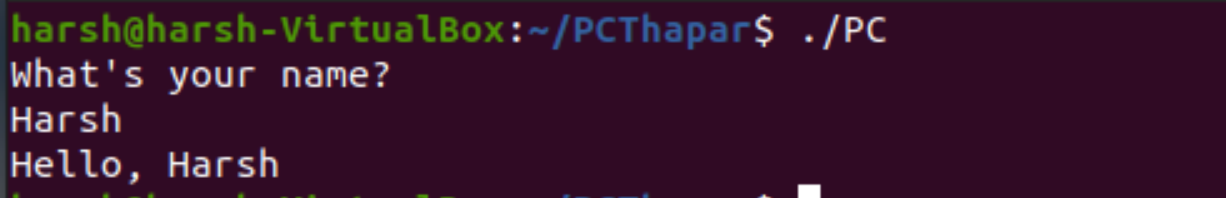
**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Question 3**

**How to ask for input in a shell script from the terminal?**

**Solution**

**To ask for input , we can prompt user simply and then read the input given by the user.**

****

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Question 4**

**How can we perform numeric comparisons in Linux?**

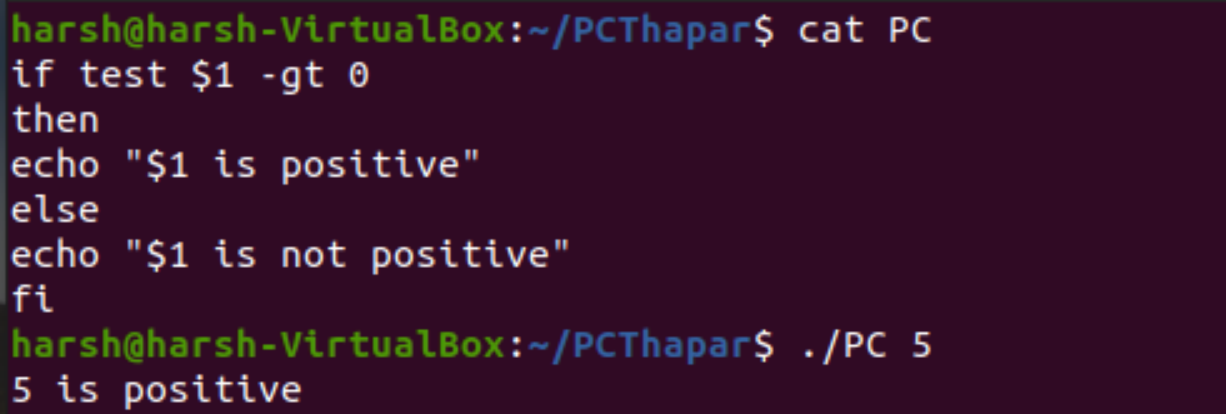
**Solution -**

Performing comparisons is done as follows.

This is one the most common evaluation method i.e. comparing two or more numbers. We will now create a script for doing a numeric comparison, but before we do that we need to know the parameters that are used to compare numerical values .

Below mentioned is the list of parameters used for numeric comparisons

* num1 -eq num2 check if 1st number is equal to 2nd number
* num1 -ge num2 checks if 1st number is greater than or equal to 2nd number
* num1 -gt num2 checks if 1st number is greater than 2nd number
* num1 -le num2 checks if 1st number is less than or equal to 2nd number
* num1 -lt num2 checks if 1st number is less than 2nd number
* num1 -ne num2 checks if 1st number is not equal to 2nd number

****

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Question 5**

**Q5. What is the syntax for different types of loops available in shell scripting?**

**Solution -**

Looping Statements in Shell Scripting: There are a total of 3 looping statements which can be used in bash programming while statement, for statement, until statement

Their descriptions and syntax are as follows:

**while statement:**

Here the command is evaluated and based on the result loop will executed, if command raise to false then loop will be terminated

**Syntax**

while command

do

Statement to be executed

done

**for statement**.

The for loop operate on lists of items. It repeats a set of commands for every item in a list. Here var is the name of a variable and word1 to wordN are sequences of characters separated by spaces (words). Each time the for loop executes, the value of the variable var is set to the next word in the list of words, word1 to wordN.

**Syntax**

for var in word1 word2 ...wordn

do

Statement to be executed

done

**until loop**

The until loop is executed as many as times the condition/command evaluates to false. The loop terminates when the condition/command becomes true.

**Syntax**

until command

do

Statement to be executed until command is true

done

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

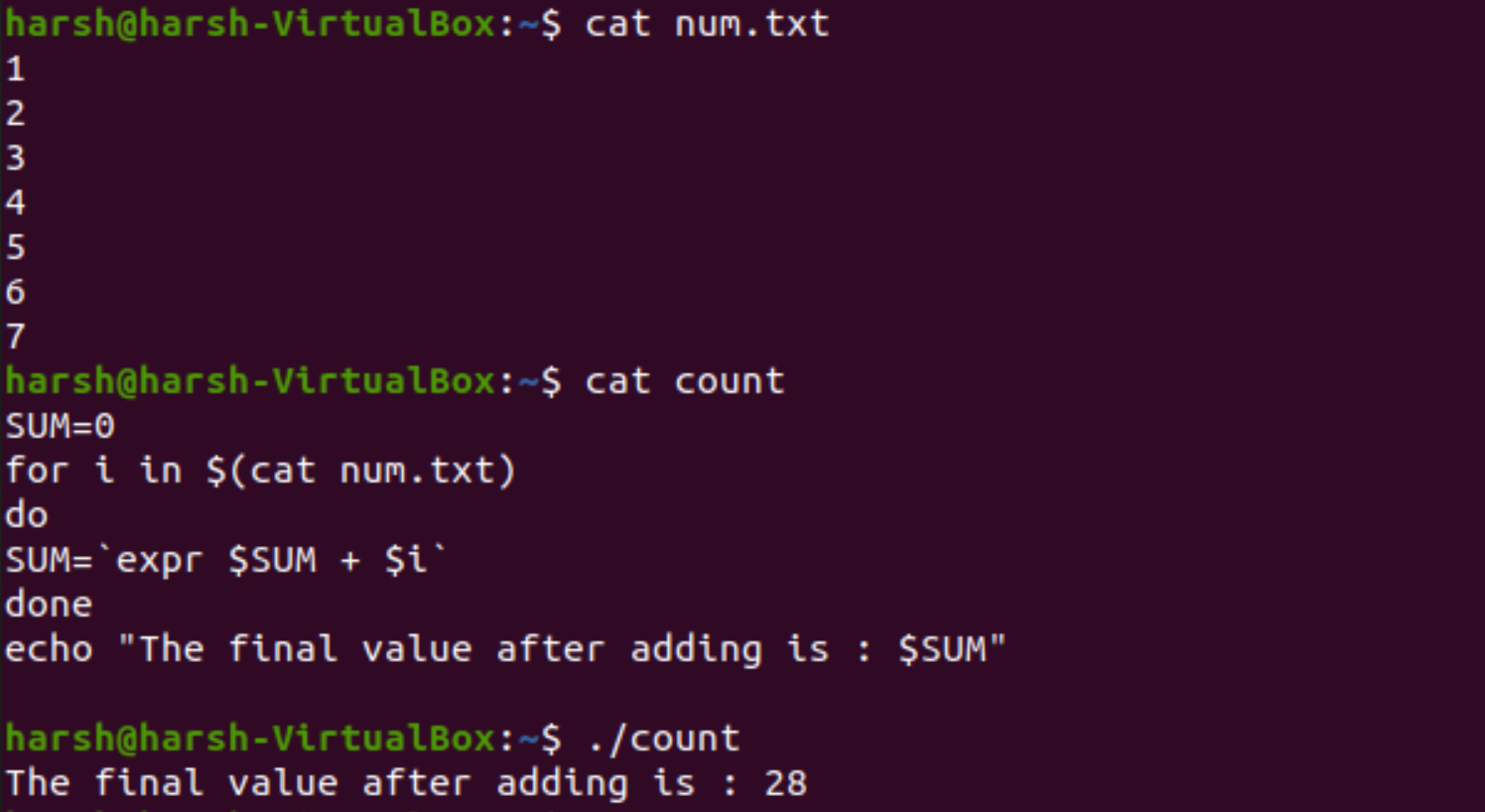
**Question 6**

**How will you find the sum of all numbers in a file in Linux?**

**Solution -**

**Create a text file which contains integers.**

**Then in other file perform the following operations.**

****

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Question 7**

**Write a shell script to validate password strength. Here are a few assumptions for the**

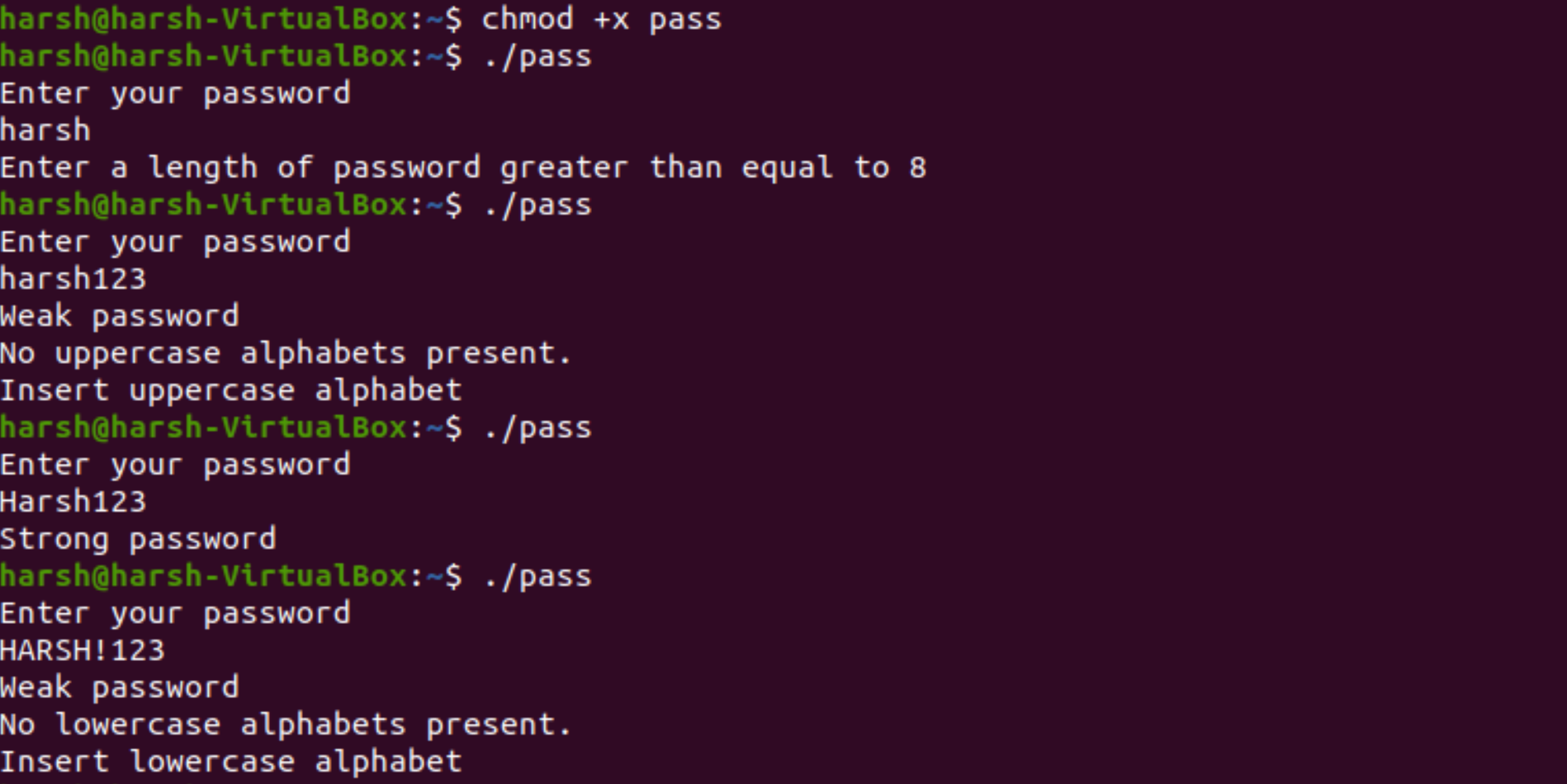
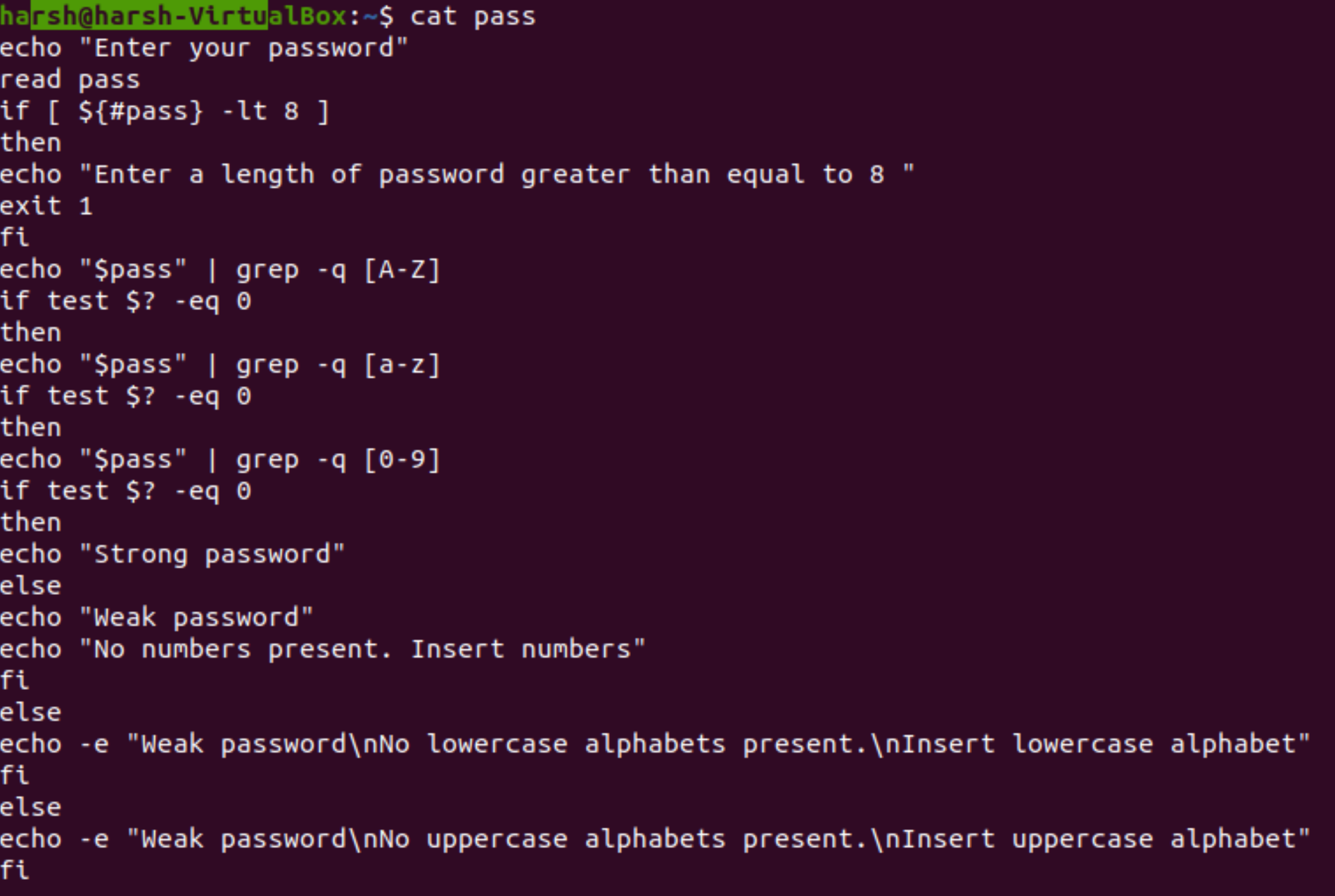
**password string.**

**• Length – a minimum of 8 characters.**

**• Contain both alphabet and number.**

**• Include both the small and capital case letters.**

**Solution -**



**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Question 8**

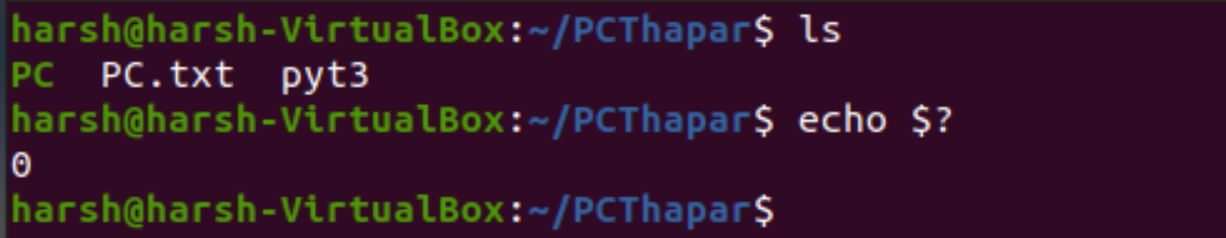
**How to check if the previous command was run successfully?**

**Solution -**

**Whenever a command runs, the return value of the command is stored in a specific bash variable.**

**Every command run in bash shell returns a value that’s stored in the bash variable “$?”. To get the value, run this command. $ echo $?**

**If a command succeeded successfully, the return value will be 0. If the return value is otherwise, then it didn’t run as it’s supposed to.**

****

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Question 9**

**How to get the last line from a file using just the terminal?**

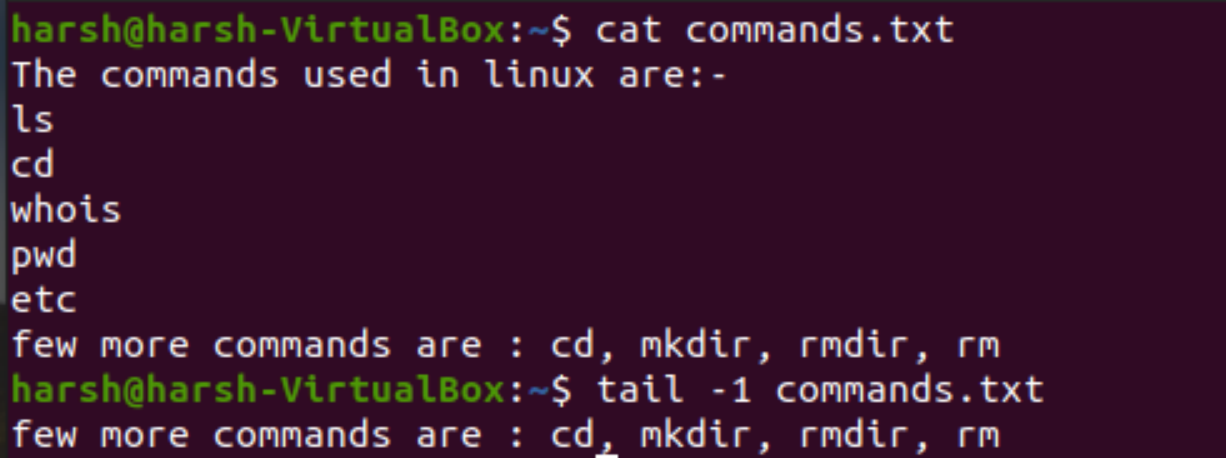
**Solution -**

We can use the tail command to see just the last line of the file.

tail [ +-[number][lbcr] ] [file]

To see last line,

tail -1 [file]



**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

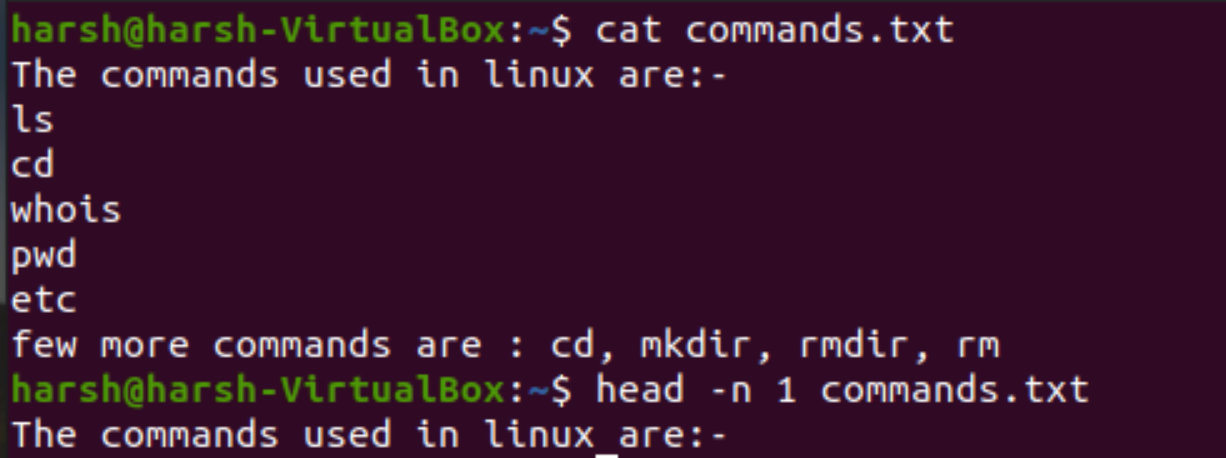
**Question 10**

**How to get the first line from a file using just the terminal?**

**Solution -**

head command is used to view the first line.

head -n 1 filename.

We put 1 to view the first line.

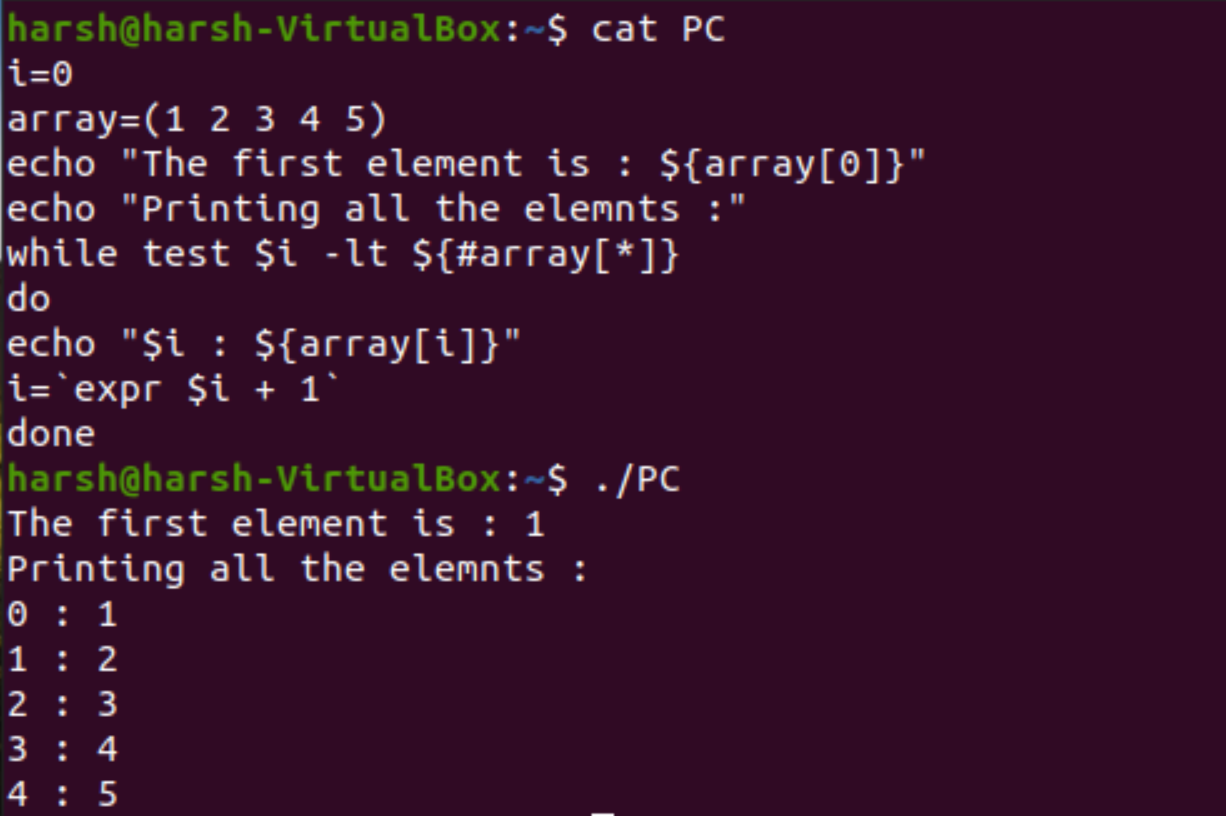
**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Question 11**

**How to print the first array element? How to print all array elements and their respective**

**indexes?**

**Solution -**



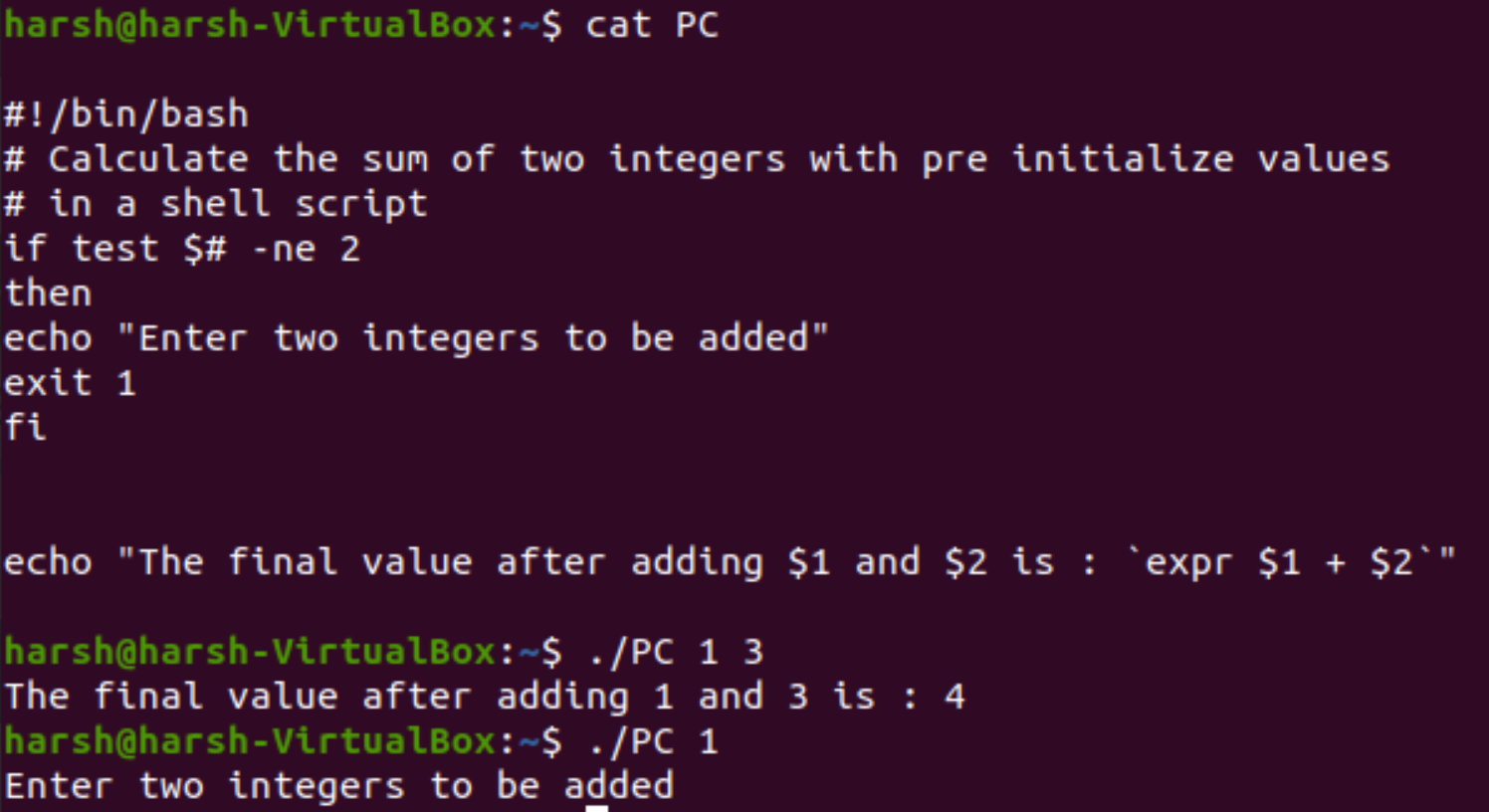
**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Question 12**

**Write a Shell Script that adds two numbers if provided as the command Line Argument and**

**if the two numbers are not entered throws an Error Message.**

**Solution -**



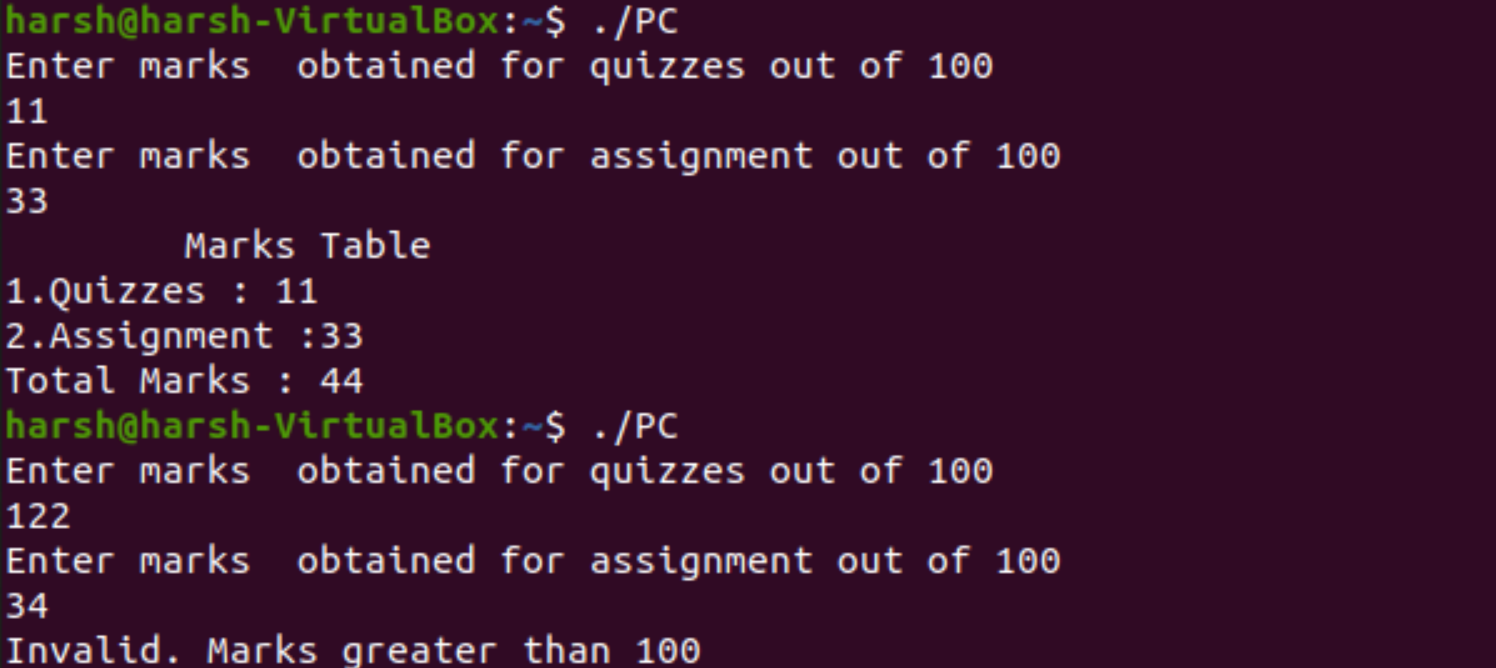
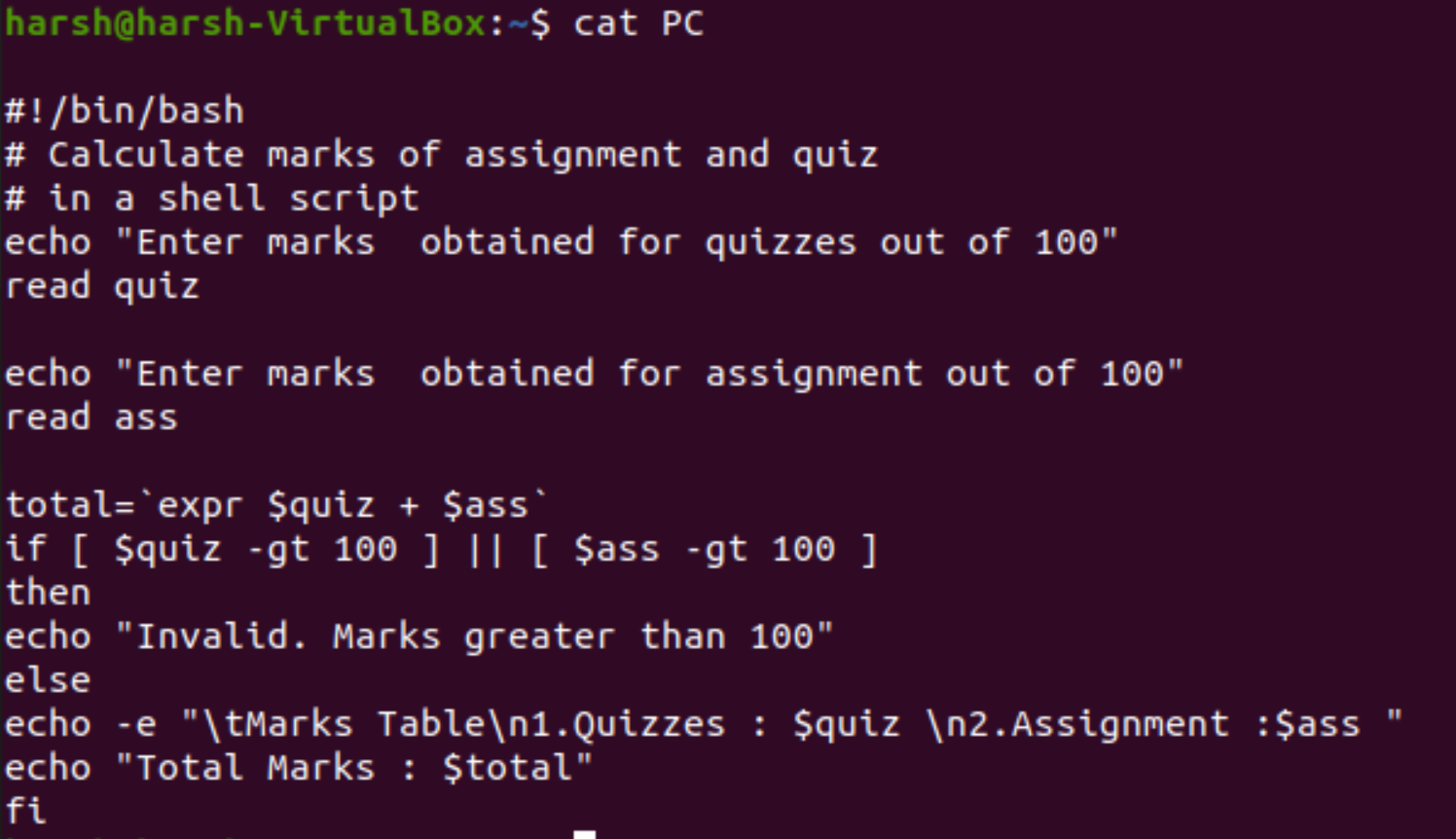
**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Question 13**

**For instance, we want to document the marks for a certain course. The total marks are 200**

**with 100 marks for Quizzes and 100 for assignments. We want to display the sum of assignments and quizzes while making sure the overall count does not exceed 200.**

**Solution -**



**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Question 14**

**Let’s take another example of a bank account program in which we want to have three**

**separate outputs for 3 different situations:**

**• The balance is less than zero**

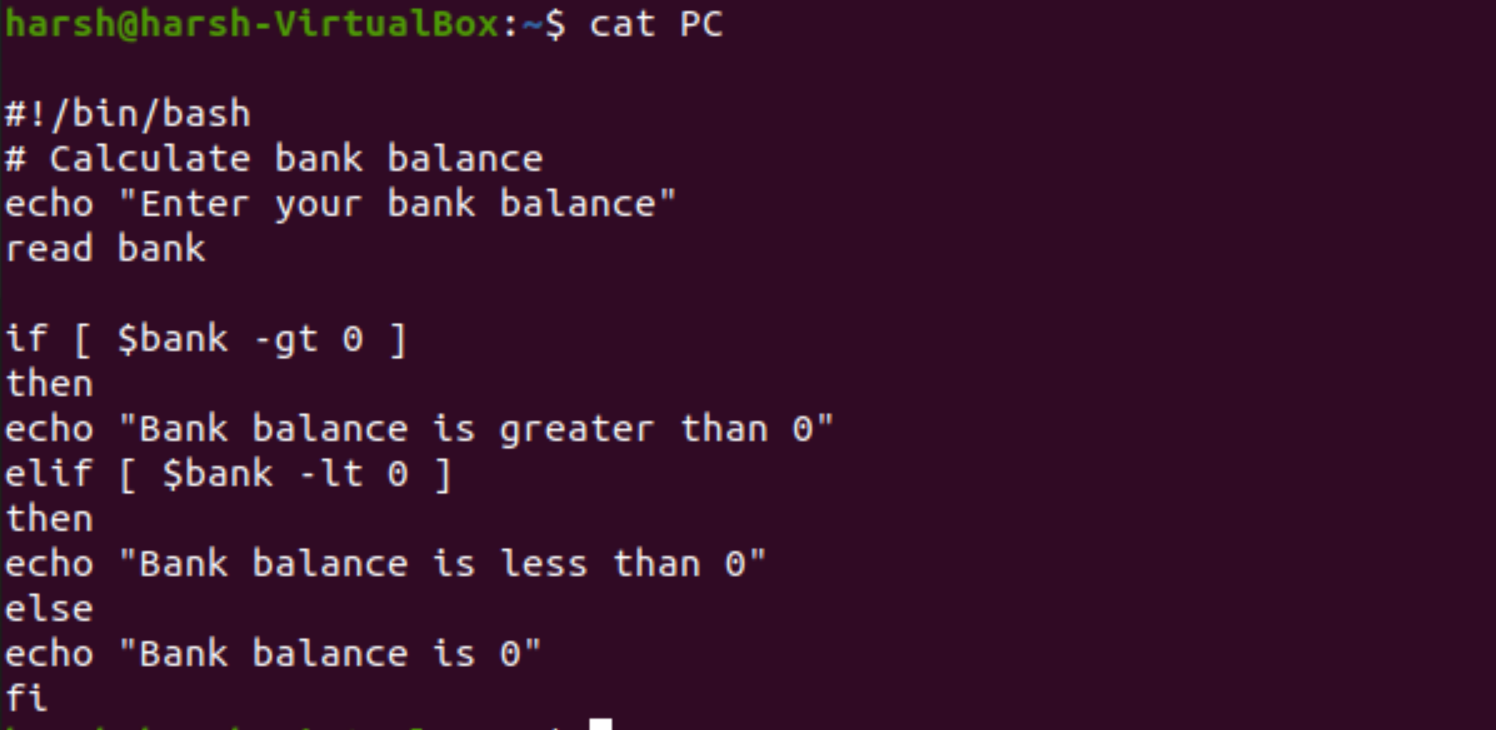
**• The balance is zero**

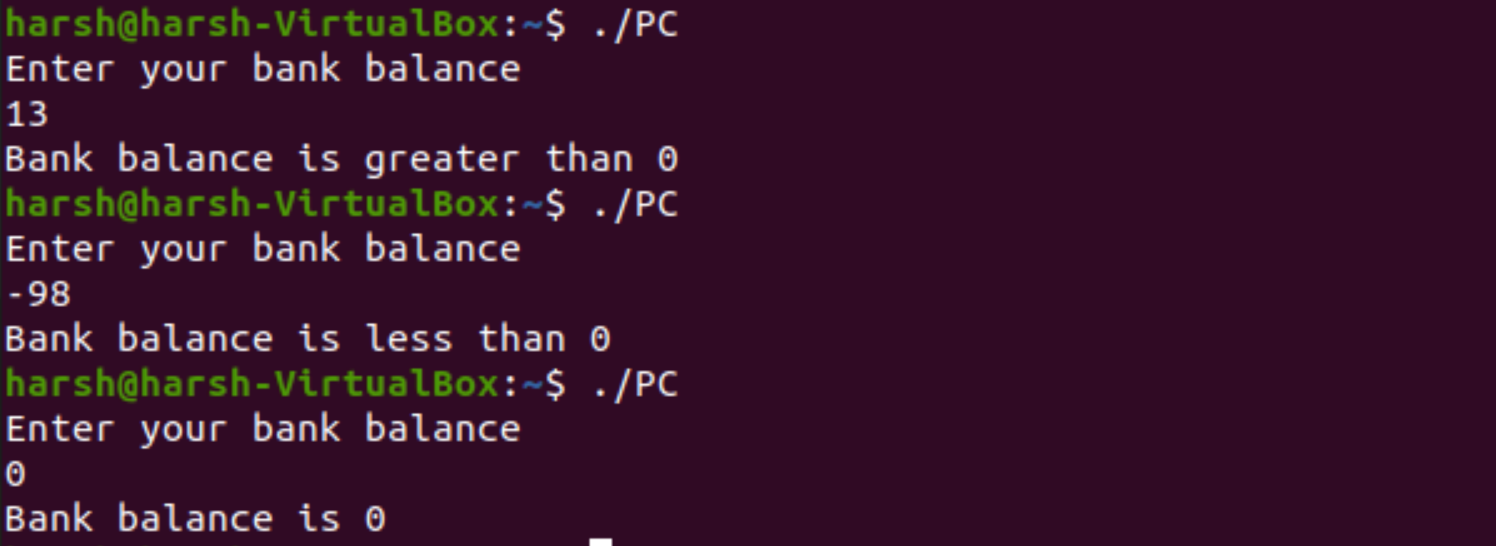
**• The balance is above zero**

**For instance, in the following program, use the if, elif, else statements to display different**

**outputs in different scenarios**

**Solution -**

****

****

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**